Network Distance Estimation with Guarantees for All Node Pairs

Aleksandrs Slivkins*

July 2006

Abstract

An active line of research in the networking community studies the distance matrix defined by the node-to-node latencies in the Internet and, in particular, provides a number of quite successful distributed approaches that approximately reconstruct these distances from observations. In such algorithms it is feasible to measure distances among only a linear or near-linear number of node pairs; the rest of the distances are simply not available. The most common framework for Internet measurements of this type is a beacon-based approach: one chooses randomly a constant number of nodes ('beacons') in the network, each node measures its distance to all beacons, and one then has access to only these measurements for the remainder of the algorithm.

To obtain theoretical insight into these recent Internet measurement studies, Kleinberg et al. [17] formulated a concrete distance reconstruction problem, termed *triangulation*, where distances from a given node to beacons form a short node label, and the unobserved distances are inferred from these labels using triangle inequality. While several significant results have been obtained in this framework, all these results include a notion of slack: they provide no guarantees for a small fraction of node pairs. Essentially, for any given positive ϵ and δ, one can reconstruct all but an ϵ-fraction of distances with multiplicative error at most $1+\delta$, using only a constant number of beacons.

In this paper we obtain triangulation-style guarantees *for all node pairs*: we reconstruct all distances with multiplicative error at most $1+\delta$, with only a poly-logarithmic load on each participating node. Our guarantees are for growth-constrained metrics, a well-studied family of metrics which have been proposed as a reasonable abstraction of Internet latencies.

1 Introduction

An active line of research in the networking community studies the distance matrix defined by the node-to-node latencies in the Internet[1] (e.g. [11, 6, 18, 31]) and, in particular, provides a number of quite successful distributed approaches that reconstruct these distances from observations [7, 24, 5, 25, 4, 20]. In such algorithms it is feasible to measure distances among only a linear or near-linear number of node pairs; the rest of the distances are simply not available. For instance, the Global Network Positioning (*GNP*) algorithm [24] uses the *beacon-based* approach where a small number of nodes ('beacons') are randomly selected in the network so that every node only measures distances to the beacons. Using only these measurements (and allowing some processing at the beacons) *GNP* empirically achieves low distortion on most node pairs.

To obtain theoretical insight into these recent Internet measurement studies, Kleinberg et al. [17] formulated the following concrete distance reconstruction problem where (following [7, 13, 11, 18]) unobserved distances are recovered using the triangle inequality. Let S be the set of beacons; and suppose for each node

*Department of Computer Science, Cornell University, Ithaca, NY 14853. *slivkins at cs.cornell.edu*.

[1]For Internet latencies the triangle inequality is not always observed; however, recent networking research indicates that severe triangle inequality violations are not widespread enough so that the node-to-node latencies can be usefully modeled by metrics.

u, and each beacon $b \in S$, we know the distance $d(u, b)$. We would like to infer the remaining unobserved distances $d(u, v)$ (when neither v nor v is a beacon), assuming we know only that we have points in an arbitrary metric space. The triangle inequality implies that

$$\max_{b \in S} |d(u, b) - d(v, b)| \leq d(u, v) \leq \min_{b \in S} (d(u, b) + d(v, b)), \tag{1}$$

and it is easy to see that these are the tightest bounds that can be provided on $d(u, v)$ if we assume only that the underlying metric is arbitrary subject to the given distances. We will say that $d(u, v)$ is reconstructed by *triangulation*[2], with distortion $\Delta \geq 1$, if the ratio between the upper and lower bounds in (1) is at most Δ.

Since it is much cheaper for nodes to exchange messages than to actually estimate their round-trip distance on the Internet (the latter typically requires a significant measurement period to produce a stable estimate), triangulation can be valuable as a way to assign each node a short *label* — its distances to all beacons — in such a way that the distance $d(u, v)$ can later be estimated by a third party (or by u or v) just from their labels.

The drawback of the beacon-based approach is the high load placed on the beacons. Indeed, the *Vivaldi* algorithm [5] and several other approaches [25, 4, 20] that followed *GNP* provide distance reconstruction algorithms with similar empirical performance where the load on *every* node is small; here the load includes computation, communication, storage and the completion time. Informally, we call such algorithms *fully distributed*. In this setting we can still *choose* beacons, but measuring distances to them directly, in the style of (1), is infeasible. However, one can hope to estimate these distances via indirect measurements, and then use these estimates to approximate the rest of the distances. Specifically, suppose for each node u and each beacon $b \in S$ we know the upper and lower bounds on $d(u, b)$, denoted respectively D^+_{ub} and D^-_{ub}. Then for any node pair (u, v) the triangle inequality implies

$$\max_{b \in S} \left(D^-_{ub} - D^+_{vb},\ D^-_{vb} - D^+_{ub} \right) \leq d(u, v) \leq \min_{b \in S} \left(D^+_{ub} + D^+_{vb} \right). \tag{2}$$

While these notions of triangulation has already lead to significant results, any performance guarantees for triangulation as defined by (1) or (2) has necessarily included a notion of *slack*, as even in very simple metrics there will be some distance pairs that cannot be reconstructed well using only a small number of beacons [17]. We say that a set of beacons achieves a triangulation with distortion $1 + \delta$ and slack ϵ if all but an ϵ fraction of node pairs in the metric are reconstructed with distortion $1 + \delta$. Such triangulations (for any given positive ϵ and δ) have been obtained in [17, 29], with poly-logarithmic number of beacons and (in the fully distributed case) poly-logarithmic per-node load. It remained an intriguing open question whether one can obtain similar triangulation-style guarantees *for all node pairs*.

Our contributions. In this paper we obtain the first triangulation-style guarantees without slack: for any given $\delta > 0$ we obtain distortion $1+\delta$ *for all node pairs*. To this end, we allow each node to have its own set of beacons. Formally, each node u acquires bounds $D^{\pm}(u, b)$ for each beacon b in some set S_b, and the $\min$ and $\max$ in (2) are taken over $S_u \cap S_b$. Instead of the number of beacons we now have the quantity $\min S_u$, termed the *order* of the triangulation. We achieve triangulations of poly-logarithmic order; our algorithm is fully distributed, with only a poly-logarithmic load on each participating node.

Our guarantees are for *growth-constrained metrics*, a well-studied family of metrics where doubling the radius of any ball increases its cardinality by at most a constant factor. Quantitatively, we define the *grid dimension* of a metric is the infimum of all α such that for any $x \geq 2$ the cardinality of any ball is at most x^α times smaller than the cardinality of a ball with the same center and x times the radius. This abstracts

[2]Note that this is one of several standard uses of the term "triangulation" in the literature; it should not be confused with the process of dividing up a region into simplices, which goes by the same name.

a useful property of d-dimensional grids (with $\alpha = d + O(1)$). Then *growth-constrained metrics* can be defined as metrics of bounded grid dimension. By definition, growth-constrained metrics can be seen as generalized grids; they have been used as a reasonable abstraction of Internet latencies in the long line of work on DHTs started by Plaxton et al. [26] (see the intro of [12] for a short survey). Growth-constrained metrics have also been considered in the theoretical computer science literature in the context of compact data structures [14], routing schemes [2], dimensionality in graphs [19], and gossiping protocols [16].

We state our main result as follows:

Theorem 1.1. *Fix an n-node metric with grid dimension α and polynomially bounded aspect ratio; suppose each node has links to 3 nodes sampled independently and near-uniformly in the network.* [3] *Then there exists a fully distributed algorithm that given any $\delta > 0$, w.h.p. computes a triangulation of order $O(\frac{1}{\delta})^\alpha (\log^2 n)$, with distortion $1 + \delta$ on all node pairs; the running time and per-node load are $O(\frac{1}{\delta})^\alpha (\log^7 n)$.*

In fact, we later show (Lemma 2.2) that instead of the initial communication graph induced by choosing three random links per node it suffices to consider any graph of low degree-expansion ratio.

Theorem 1.1 is also meaningful for *Internet Tomography* (e.g. see [15]), in a setting where the network measurements are reported to and processed at a central location. In this setting treat the network as an oracle which for any given node pair (u, v) returns $d_{\mathrm{M}}(u, v)$ at unit cost to both u and v. We (essentially) do not need to worry about communication and processing, yet it is still desirable, and quite non-trivial, to reduce the measurement load on nodes.

A crucial element of our construction is *rings of neighbors*, a sparse distributed data structure which captures the distance information in the network. The idea is that every node u stores pointers to some nodes called 'neighbors'; these pointers are partitioned into several 'rings' so that the neighbors in the i-th ring are selected near-uniformly in a ball of radius 2^i around u. In effect, rings of neighbors form an overlay network with a certain structure imposed by the rings.[4] Rings of neighbors can be used as a framework for location-aware network applications; in particular, it is essentially the data structure that underlies *Meridian* [32], a system for location-aware node selection in a network. A similar data structure was explored in [28] in the context of off-line node labeling problems. Accordingly, our fully distributed construction for rings of neighbors is interesting on its own right, by way of providing provable guarantees for *Meridian* and as a stepping stone for further algorithmic work. Parts of this construction build on (a much cleaner version of) the ideas from [29].

Related work and further directions. Work on distance labeling [9] seeks to assign a short label to each node in a graph so that the distance between u and v can be (approximately) determined from their labels alone. This is of course analogous to our goals in triangulation. In particular, [30, 29, 22, 28] investigated $(1 + \delta)$-approximate distance labels for doubling metrics, a common generalization of growth-constrained metrics and constant-dimensional Euclidean metrics. Both the objective and the techniques in this line of research differ considerably from our work here: the concern is with labels of low bit complexity, but the encoding of distances into short labels makes extensive use of the full distance matrix, and thus is not adaptable to our distributed setting in which only a small fraction of distances can be measured.

In [24, 5] and several subsequent papers the unobserved distances are reconstructed in the form of an embedding into a low-dimensional Euclidean space. Motivated by these empirical approaches, [17, 29, 1] provided distributed metric embeddings with provable guarantees, both in the beacon-based and in the fully distributed frameworks (for arbitrary and doubling metrics, respectively). These guarantees, like the previously obtained guarantees for triangulation, had slack in them: they allowed for arbitrarily small

[3] *Aspect ratio* of a metric is the ratio between the largest and the smallest distance. We say that a distribution τ is *near-uniform* if $\|\sigma_{\text{unif}} - \tau\|_\infty \leq \frac{1}{2n}$, where σ_{unif} is the uniform distribution.

[4] Note that the term 'neighbor' here refers to the adjacency in this overlay network, not to the proximity in the input metric.

distortion on a small fraction of node pairs. It would be very interesting to extend the techniques from this paper to distributed embeddings with guarantees for all node pairs.

Preliminaries. We assume that nodes do not share data and communicate via messages. Specifically, each node follows the following cycle: receive a message, do local computation, (possibly) send messages to other nodes, wait for the next message received. Each node has an *address* that other nodes need to know in order to contact this node. These addresses cannot be guessed; they can either be given in advance or passed from one node to another. In particular, initially every node is given a (possibly empty) list of addresses. Later in the algorithm, a node may delete some addresses from its list, or may add some new addresses received from other nodes. The nodes whose addresses are currently in the list of node u are called the *neighbors* of u. For simplicity we assume that each address takes $O(1)$ space.

Let G be a graph, directed or undirected, possibly with multiple edges and self-loops. Say an algorithm is *G-distributed* if it conforms to the above model so that initially every node u is given the addresses of all its G-out-neighbors (if G is directed) or of all its G-neighbors (if G is undirected), together with the corresponding multiplicities. We call G the *initial communication graph*.

The *load* on a given node includes computation, storage, and communication. For simplicity the load is defined as a sum $x + y + z$, where x is the number of CPU cycles, y is the number of bytes used for storage, and z is the number of bits sent and received; we will use the $O(\cdot)$ notation, so the exact units do not matter. The load of an algorithm is the maximal load on a node. If the algorithm starts at time 0, and terminates at time τ_u on each node u, then the *total running time* is defined as $\max \tau_u$. Note that it can be very different from the load, since the latter in general does not include the idle time.

Our constructions are highly randomized, and our guarantees are *with high probability*, which in this paper means that the failure probability is at most $1/n^c$, for a sufficiently high constant c.

Map of the paper. We start with a section on distributed random walks, which are one of the principal tools used in this paper. We construct rings of neighbors in Section 3. Finally, in Section 4 we use rings of neighbors to obtain the desired triangulation. Some relevant background on probability and expander graphs is covered in Appendix A.

2 Tools: distributed random walks

In this section we discuss load-balanced random node selection via distributed random walks.

Let n be the cardinality of V, and let σ_{unif} be the uniform distribution on V. Say a distribution τ on V is *near-uniform* if $\|\sigma_{\text{unif}} - \tau\|_\infty \leq \frac{1}{2n}$. We can define near-uniform distributions on any given subset of nodes in a similar fashion. Say random variables $X_1 \ldots X_k$ are *Q-nice* if their joint distribution is that of k independent random variables with a near-uniform distribution on Q.

We start with a fact on Markov chains which will be essential to our constructions. Consider an undirected graph $G = (V, E)$. Let λ_{uv} be the multiplicity of edge uv, and let $d_u = \sum_v \lambda_{uv}$ be the degree of node u. For any $d \geq \deg(G)$ let us define the Markov chain $M_{(G,d)}$ as follows:

$$M_{(G,d)}(u,v) = \begin{cases} \lambda_{uv}/2d & \text{if } u \neq v \\ 1 - (d_u - \lambda_{uv})/2d & \text{otherwise} \end{cases} \tag{3}$$

It is easy to see that this Markov chain has a uniform stationary distribution. Moreover, by Theorem A.4 for graphs of low degree-expansion ratio (see Appendix A) it has a rapid mixing property:

Lemma 2.1. *Let G be a connected undirected graph (possibly with loops and parallel edges) with expansion γ. Then for any $d \geq \deg(G)$ and $k \geq O(d/\gamma)^2(\log n)$ the k-step distribution of $M_{(G,d)}$ is near-uniform for any initial distribution.*

Proof. Let $M = M_{(G,d)}$. Note that M is irreducible since G is connected, and M is aperiodic since every node has a positive stalling probability. Therefore M is ergodic. M is time-reversible since $M(u,v) = M(v,u)$ holds for all node pairs. Since $M(u,v) \geq \lambda_{uv}/2d$ for all node pairs, the expansion of M (as an edge-weighted graph) is γ/d. Now the Lemma follows from Theorem A.4. □

In the following result each node runs its own copy of the random walk from Lemma 2.1.

Lemma 2.2. *Let $G = (V,E)$ be an undirected graph on n nodes, of expansion γ. Suppose numbers $d \geq \deg(G)$ and $t \geq (d/\gamma)^2(\log n)$ are known to all nodes. Then for any $k \in \mathbb{N}$ there exists a randomized G-distributed algorithm whereby every node u acquires k addresses X_{uj}, $j \in [k]$, such that $\{X_{uj} : u \in V, j \in [k]\}$ are V-nice random variables in the probability space induced by the algorithm. The running time and (with high probability) the load are $O(t \times \max(k, \log n))$; the per-node storage is $O(k+d)$.*

Proof. By abuse of notation, let us fix some enumeration f of V and treat each node u as a unit vector in the $f(u)$-th dimension. Let σ_{unif} be the uniform distribution on V.

For a node v, let $\mathcal{A}_v = \mathcal{A}_v(G,d)$ be a (v,G)-distributed algorithm that starts at v and simulates the Markov chain $M_{(G,d)}$ for t steps. Specifically, at every step i the Markov chain visits some node X_i, which means the following: node X_i selects one of its G-neighbors (or itself) according to the distribution (3) and forwards the Markov chain to this node. The process starts at $X_0 = v$, and terminates at step t by returning the value X_t to node v.

Note that by Lemma 2.1 X_t is a random variable with a near-uniform distribution. For simplicity let us assume that at each step i nodes v and X_i experience a unit load each. It follows that for a given node w the expected load induced by algorithm $\mathcal{A}_v$, $v \neq w$ is equal to

$$\Pr[X_i = w] = \left(M^{(i)}_{(G,d)} v\right) \cdot w. \tag{4}$$

The overall algorithm is simple: every node u initiates k independent copies of algorithm $\mathcal{A}_u$. In the course of this algorithm, each message processed by a given node u is related to a certain step of some $\mathcal{A}_v$. To simplify the analysis of the total running time, let us assume that whenever there is contention, messages from earlier steps are given higher priority.

Note that the per-node storage requirement is $O(k+d)$, since at any point in time a given node u needs to store only the addresses of all his G-neighbors, the current step for each of the k copies of algorithm $\mathcal{A}_u$.

Let us fix a node w and a step $i \in [t]$. Let Y_{vj} be the load induced on w by the j-th copy of algorithm $\mathcal{A}_v$. Then by (4) we have

$$\sum_{v \neq w} E\left(Y_{vj}\right) = \sum_{v \neq w} \left(M^{(i)}_{(G,d)}\, v\right) \cdot w \leq n \left(M^{(i)}_{(G,d)}\, \sigma_{\text{unif}}\right) \cdot w = O(n\, \sigma_{\text{unif}} \cdot w) = O(1).$$

Since $\{Y_{vj}\}$ is a family of bounded independent random variables, by Chernoff Bounds (Theorem A.1b) with $\mu = \Theta(\max(k, \log n))$ it follows that $\sum_{\text{all } (v,j)} Y_{vj} \leq 2\mu$ with high probability. In particular, the total load on any given node (over all steps) is $O(t\mu)$ with high probability.

To bound the total running time, we claim that the processing of each step i completes, for all nodes, by time $O(i\mu)$ with high probability. Indeed, suppose a given step i is complete by time $O(i\mu)$. Since with high probability every given node u needs to process at most $O(\mu)$ messages for step $i+1$, and these messages have priority over those from later steps, processing them will take at most $O(\mu)$ time. Claim proved. □

By the above result, in Theorem 1.1 and Theorem 3.2 instead of the initial communication graph induced by choosing three random links per node it suffices to consider any graph of low degree-expansion ratio. We also use Lemma 2.2 to construct the out-most rings of neighbors in Section 3. For all other rings we need a more general version of this lemma, where each node u samples from some subset Q_u. To use such result, we need to bound the expected load on all nodes in $Q_u \setminus \{u\}$ by a small multiple of $1/|Q_u|$.

Lemma 2.3. *Let $G = (V, E)$ be an undirected graph on n nodes. Fix node u and consider a subset $Q \subset V$ such that the graph $G|Q$ has expansion γ. Suppose that:*

- *after pinging any node $v \in V$, node u can, at unit cost, determine whether $v \in Q$.*
- *node u knows numbers $d \geq \deg(G)$, $d_Q \geq \deg(G|Q)$ and $t \geq (d_Q/\gamma)^2 (\log n)$,*
- *node u is given a* random seed*: an address of some node.*

Then for any $k \in \mathbb{N}$ there exists a randomized G-distributed algorithm (initiated by u) such that:

(a) node u acquires addresses of k nodes $X_i \in Q$, where the X_i's are Q-nice random variables. The running time and the load on node u are $O(kdt)$.

(b) The load on every other node w is at most $O\left(\sum_{wv \in G} Z_v\right)$, where Z_v is the number of times node v is "visited" by the algorithm,[5] which is at most kt for all $v \in Q$, and 0 otherwise. If the random seed was selected independently from a near-uniform distribution τ on Q, then in the probability space induced by the algorithm and τ, $E(Z_v) = O(kt/|Q|)$ for each $v \in Q$.

Proof. We use algorithm $\mathcal{A}_v(G|Q,\, d)$ defined in the proof of Lemma 2.2, in a slightly modified form. Specifically, at each step i of this algorithm node u communicates with some node $X_i \in Q$, asks this node for a list of its G- neighbors, determines which of these neighbors lie in Q, and chooses the next node X_{i+1} among those according to the distribution $M_{(G|Q,\, d)}$, see (3). The process starts at $X_0 = v$, and terminates at step t by returning X_t to node u. During each step node u incurs load $O(d)$, and node X_i incurs load $O(1)$.[6] Given this modified form of $\mathcal{A}_v(G|Q,\, d)$, the overall algorithm is simple: node u initiates k independent copies of algorithm $\mathcal{A}_w(G|Q,\, d)$, where w is the given random seed.

Parts (a) is trivial. For part (b), we define Z_v to be the number of times some copy of algorithm $\mathcal{A}_v(G|Q,\, d)$ selects node v as the next step. Let us fix some node $v \in Q \setminus \{u\}$, and let Y_{ij} be the number of times node v is visited by the i-th step of the j-th copy of the random walk. Let σ_{unif} be the uniform distribution on Q, and let $M = M^{(i)}_{(G|Q,\, d)}$ be the i-th power of the corresponding transition matrix. Note that $M\sigma_{\text{unif}} = \sigma_{\text{unif}}$, so rows of M have unit sums, so $\|M\tau\|_\infty \leq \|\tau\|_\infty = O(1/|Q|)$. Consider the probability space induced by the algorithm and τ. Then

$$E[Y_{ij}] = O\left(\Pr_\tau[X_i = v]\right) = O\left((M\,\tau) \cdot v\right) = O(1/|Q|).$$

We get $E(Z_v) = O(kt/|Q|)$ by summing over all $i \in [t]$ and $j \in [k]$. □

3 Randomized Rings of Neighbors

In this section define and construct *rings of neighbors*; to this end we will extensively use distributed random walks, namely Lemma 2.2 and Lemma 2.3. In what follows, let us fix a metric d_{M} on an n-node set V, with grid dimension α and polynomially bounded aspect ratio. Let $B_u(r)$ be the closed ball of radius r around node u, i.e. $B_u(r) = \{v \in V : d_{\text{M}}(u, v) \leq r\}$. Without loss of generality assume that the minimal distance is 1; let Δ be the diameter of the metric. Denote $B_{ui} = B_u(\Delta/2^i)$.

[5] For each node v, the algorithm either does not touch the list of its G-neighbors, or *visits* v: reads the entire list at once.

[6] Node X_i sends a list of d addresses. However, in practice this list should fit in a very small number of packets.

We seek to construct a distributed data structure that we call *rings of neighbors*. In this data structure, for each $i \in [\log \Delta]$ each node u stores addresses of k other nodes in B_{ui}. We denote these k nodes as $X_u^{(i)} = \left\{ X_{uj}^{(i)} : j \in [k] \right\}$ and call $X_u^{(i)}$ the *i-th ring of neighbors* of node u. Here k is a small number, e.g. $k = \Theta(\log n)$, which we call *ring cardinality*.

Suppose we have a randomized algorithm which constructs the rings of neighbors and, consequently, induces a joint probability distribution on random variables $\{X_{uj}^{(i)}\}$. Intuitively, we would like these random variables to be independent and uniformly distributed on the respective balls B_{ui}. We will construct a slightly weaker version. Specifically, let $\mathcal{F}_i$, $i \in [\log \Delta]$ be the collection

$$\mathcal{F}_i = \left\{ X_{uj}^{(i)} : u \in V, j \in [k] \right\}. \tag{5}$$

For notational convenience, define $\mathcal{F}_{-1}$ to be empty. We will construct rings of neighbors such that given $\cup_{l<i} \mathcal{F}_l$ all random variables in $\mathcal{F}_i$ are conditionally independent and near-uniformly distributed.

Definition 3.1. Consider a metric space (V, d_{M}) on n nodes, with aspect ratio Δ. *Randomized Rings of Neighbors* (RRN) on this metric space is a joint distribution on V-valued random variables

$$\left\{ X_{uj}^{(i)} : u \in V, i \in [\log \Delta], j \in [k] \right\}$$

such that with high probability the following two properties hold for each $i < \lceil \log \Delta \rceil$:

(P1) given $\cup_{l<i} \mathcal{F}_l$, random variables in $\mathcal{F}_i$ are conditionally independent.

(P2) given $\cup_{l<i} \mathcal{F}_l$, each random variable $X_{uj}^{(i)} \in \mathcal{F}_i$ has a near-uniform distribution on B_{ui}.

Here k is a fixed parameter called *ring cardinality*, and $\mathcal{F}_i$ is defined by (5).

We provide a fully distributed construction for Randomized Rings of Neighbors on a growth-constrained metric with initial communication graph of low degree-expansion ratio.

Theorem 3.2. *In the setting of Theorem 1.1, for any given constant $c \geq 1$ there exists a randomized G-distributed algorithm with running time and load $O(c^4)\, 2^{O(\alpha)} (\log^7 n)$ which with high probability constructs Randomized Rings of Neighbors on (V, d_M) with ring cardinality $\Omega(c\, 4^\alpha \log n)$.*

Overview of the algorithm. Our construction proceeds in $\lceil \log \Delta \rceil$ stages. A given stage $i \in [\log \Delta]$ handles distances on the scale of $\Delta/2^i$: this is when all $(i+1)$-th ring neighbors are constructed. Specifically, at the beginning of stage i, the i-th ring neighbors of all nodes have already been constructed. For each node u, they induce a low-degree expander Q on the ball $B_{(u,i+1)}$; this is essentially because they are conditionally independent and distributed near-uniformly on the corresponding balls of radius $\Delta/2^i$. Node u selects its $(i+1)$-th ring neighbors by executing independent random walks on Q; by the expansion property of Q in order to guarantee near-uniformity it suffices to run these random walks for poly-log many steps. In fact, we use Lemma 2.3 for the random walks. Note that we need to be careful to separate the random seeds used in this lemma from the graph on which we do the random walk. For every node a large portion of the load comes from helping other nodes choose their neighbors; one needs to guarantee that no node is overloaded by helping others.

In the rest of this section we prove Theorem 3.2. We use the grid dimension via a simple corollary:

Lemma 3.3. *Suppose d_M is a metric with grid dimension α. Fix any two nodes u, v and let $d = d_M(u, v)$. Then for any positive r, r^* such that $\frac{d+r}{r^*} \geq 2$ we have $|B_u(r)| \leq (\frac{d+r}{r^*})^\alpha |B_v(r^*)|$.*

Let us set $k = c\,4^\alpha \lceil \log n \rceil$ to be the ring cardinality, where the constant c is chosen at least large enough so that the following property holds:

(P3) Consider any two balls $B_{(v,\,i+1)} \subset B_{ui}$. Suppose k nodes are chosen independently from a near-uniform distribution on B_{ui}. Then with high probability at least $3 \log n$ of them land in $B_{(v,\,i+1)}$.

Note that in (P3) $|B_{ui}| \le 4^\alpha |B_{(v,\,i+1)}|$ by Lemma 3.3, so $c = O(1)$ does indeed suffice.

Recall that the rings of neighbors are described by random variables $X_{ui}^{(i)}$, which are grouped in collections $\mathcal{F}_i$, see (5). We start by constructing $\mathcal{F}_0$ using Lemma 2.2 applied to the original connectivity graph G. Such $\mathcal{F}_0$ clearly satisfies conditions (P1) and (P2). Since G is a $O(\log n)$-degree expander, in Lemma 2.2 we take $t = O(\log^3 n)$, so for stage 0 the running time and (with high probability) the load are $2^{O(\alpha)}(\log^4 n)$, and the storage requirement is $O(k)$.

The rest of the construction proceeds in stages, so that in stage $i \ge 0$ we construct $\mathcal{F}_{i+1}$ assuming that we have already constructed $\mathcal{F}_i$ that satisfies (P1) and (P2). Let us partition the family $\mathcal{F}_i$ of random variables into two subfamilies:

$$\mathcal{F}_i^{\text{walk}} = \left\{ X_{uj}^{(i)} :\ u \in V,\, j \in [k/2] \right\} \ \text{ and } \ \mathcal{F}_i^{\text{seed}} = \mathcal{F}_i \setminus \mathcal{F}_i^{\text{walk}}.$$

We will invoke Lemma 2.3, independently for every node u. The underlying graph for the random walks will come from $\mathcal{F}_i^{\text{walk}}$, and the random seeds will come from $\mathcal{F}_i^{\text{seed}}$. It is important that the random seed is independent of $\mathcal{F}_i^{\text{walk}}$ (conditionally, given $\mathcal{F}_{i-1}$).

Let us define G_i^* to be the directed graph induced by $\mathcal{F}_i^{\text{walk}}$, namely a directed graph on V, possibly with self-loops and multiple edges, which contains an edge uv whenever $v = X_{uj}^{(i)}$ for some $j \in [k/2]$. Let G_i be the undirected version of G_i^*. In proactive, to construct G_i each node u just contacts all of its $\mathcal{F}_i^{\text{walk}}$-neighbors to let them know that they should store a link to u. Note that G_i has a low degree:

Claim 3.4. $\deg(G_i) \le O(k\,2^\alpha)$ *with high probability.*

Proof. Condition on $\mathcal{F}_{i-1}$ and consider the probability space induced by $\mathcal{F}_i^{\text{walk}}$. For a given node u, it suffices to bound its in-degree in G_i^*. Note that $vu \in G_i^*$ only if $u \in B_{vi}$ or, equivalently, $v \in B_{ui}$. Each node $v \in B_{ui}$ has k links distributed near-uniformly on B_{vi}. Each of these links lands in u with probability at most $2/|B_{vi}|$, which is at most $2^\alpha/|B_{ui}|$ by Lemma 3.3. The expected in-degree of u in G_i^* is thus at most $k\,2^\alpha$. The claim follows by Chernoff Bounds since by (P1) all links in G_i^* are independent given $\mathcal{F}_{i-1}$. □

For a given node u, let us define $Q_u = B_{(u,\,i+1)}$. We analyze the induced graph $G_i|Q_u$:

Claim 3.5. *The induced graph $G_i|Q_u$ is an $O(k)$-degree expander with high probability.*

Proof. Condition on $\mathcal{F}_{i-1}$ and consider the probability space induced by $\mathcal{F}_i^{\text{walk}}$. Each node $v \in Q_u$ has k out-links in G_i^*. Since $Q_u \subset B_{vi}$, by (P2) each of these links lands into a given node $w \in Q_u$ with probability at most $2/|Q_u|$. The expected in-degree of w in $G_i^*|Q_u$ is thus $O(k)$. Since by (P1) all links in G_i^* are independent given $\mathcal{F}_{i-1}$, by Chernoff Bounds the in-degree of $G_i^*|Q_u$ is at most $O(k)$ with high probability, and consequently so is the degree of $G_i|Q_u$. Moreover, by (P3) with high probability the out-degree of $G_i^*|Q_u$ is at least $3 \log n$, so by Theorem A.3 with high probability $G_i|Q_u$ is an expander. □

By (P3) with high probability for each node u at least $3 \log n$ nodes in $\mathcal{F}_i^{\text{seed}}$ lie inside Q_u. Pick one such node at random, denote it Y_u. For a given node u, let $\mathcal{A}_u$ be the construction in Lemma 2.3 whereby node u acquires the addresses of k near-random nodes. Specifically, we invoke this construction for subset $Q = Q_u$, underlying graph G_i, random seed Y_u, and (by Claims 3.4 and 3.5) upper bounds

$$d = O(k\,2^\alpha) \text{ and } d_Q = O(k) \text{ and } t = O(k^2 \log n).$$

The overall construction for stage i is simple: each node u invokes algorithm $\mathcal{A}_u$ and thereby acquires the addresses of k nodes in Q_u, not necessarily distinct. Define $X_{uj}^{(i+1)}$ to be the j-th of these nodes. Clearly properties (P1) and (P2) are satisfied. It remains to bound the per-node load.

Let Z_{vu} be the quantity from Lemma 2.3(b), the number of times node v is "visited" by algorithm $\mathcal{A}_u$. Recall that $Z_{vu} = 0$ whenever $v \notin Q_u$ or, equivalently, when $u \notin Q_v$. Let us define $Z_v = \sum_{u \in V} Z_{vu}$, the total number of times node v is visited by some $\mathcal{A}_u$. Let us bound Z_v:

Claim 3.6. *Z_v is at most $O(kt\,2^\alpha)$ in expectation, and at most $O(kt\,2^\alpha \log n)$ with high probability.*

Proof. Let us condition on $\mathcal{F}_{i-1}$ and $\mathcal{F}_i^{\text{walk}}$ (i.e. let us assume that those are fixed), and let us consider the probability space induced by the random choices in $\mathcal{F}_i^{\text{seed}}$ and in algorithms $\{\mathcal{A}_u : u \in V\}$. By Lemmas 2.3(b) and 3.3, for each $u \in Q_v$ we have

$$E(Z_{vu}) \leq O(kt/|Q_u|) \leq O(kt\,2^\alpha)/|Q_v|,$$

so $E(Z_v) \leq O(kt\,2^\alpha)$. Since the random variables $\{Z_{vu} : u \in Q_v\}$ are independent, the claim follows by Chernoff Bounds. □

Let us fix some node w and partition the total load experienced by node w in a given stage into *direct load* induced on w by algorithms $\mathcal{A}_w$, and *indirect load* induced on w by algorithms $\mathcal{A}_u$, $u \neq w$. By Lemma 2.3(b) the direct load on node w is $O(kdt)$, and the indirect load on w is $O\left(\sum_{wv \in G_i} Z_v\right)$. By Claims 3.4 and 3.6, the latter is at most $O(k^2 t\,4^\alpha)$ in expectation, and at most $T = O(k^2 t\,4^\alpha \log n)$ with high probability; the latter is at most $O(2^{10\,\alpha} \log^6 n)$. Summing over all stages, the total load is $O(T \log n)$ with high probability.

Let us bound the running time for a given stage. Recall that each message belongs to a particular step of one of the random walks. To simplify the analysis, let us assume that whenever there is contention, messages from earlier steps are given higher priority, and among messages from the same step, a given node u gives higher priority to messages related to algorithm $\mathcal{A}_u$. Via the same analysis as above we can show that during each step a given node receives at most T/t messages. It follows that in time $O(T/t)$ a given node u receives "answers" to all messages sent by a given step of algorithm $\mathcal{A}_u$. Therefore the total running time for a given stage is at most $O(T)$, as required. This completes the proof of Theorem 3.2.

4 Network Triangulation: Proof of Theorem 1.1

Overview of the algorithm. As in the previous section, our construction proceeds in $\lceil \log \Delta \rceil$ stages so that each stage $i \in [\log \Delta]$ handles distances on the scale of $\Delta/2^i$. First each node selects itself (independently at random) as a *level-i beacon*; we make sure that level-i beacons are sufficiently dense on the scale of δr, and yet sufficiently sparse on the scale of r. Then level-i beacons declare themselves to other nodes via a special broadcast, so that each node (a) finds out about the nearby level-i beacons, (b) forms upper and lower bounds on distances to these beacons. These bounds are not necessarily precise enough to guarantee a sufficiently accurate triangulation. Thus we need an essential *update step* where each level-i beacon b updates the distance estimates to all level $j \leq i-2$ beacons that it knows; this is accomplished by querying all level-$(i-1)$ beacons that b is aware of. For every node a large portion of the load comes from helping other nodes form their estimates. As in the previous section, one needs to be very careful to guarantee that no node is overloaded by helping others.

We keep all notation from Section 3; we further need the ring cardinality k to satisfy the following:

(P4) Pick any r and any two nodes u, v such that $d_{\text{M}}(u,v) \leq \frac{4}{3}\,r$. Choose k nodes independently from a near-uniform distribution on $B_u(2r)$. Then with high probability at least one of them lands in $B_v(\frac{2}{3}\,r)$.

(P5) Pick any r and any node u. Suppose each node $v \in B_u(r)$ is selected k times, independently, with probability at least $1/2|B_v(r)|$. Then with high probability at least one node in $B_u(r)$ is selected.

These properties is very similar to (P3), but are slightly different quantitatively. By Lemma 3.3 in (P5) we have $|B_v(r)| \leq 2^\alpha |B_u(r)|$, so by Chernoff bounds the minimal k such that (P5) holds is $k_{\text{b}} = O(2^\alpha \log n)$; we will use this quantity k_{b} later in the proof. Similarly, in (P4) we have $|B_v(\frac{2}{3}\,r)| \leq 5^\alpha |B_u(\frac{4}{3}\,r)|$, so $k = O(5^\alpha \log n)$ suffices.

Consider the construction of RRN in Theorem 3.2, for the ring cardinality $k = c\,5^\alpha \lceil \log n \rceil$, where the constant c is chosen at least large enough so that all three properties (P3), (P4) and (P5) hold. We describe the RRN by a collection of directed graphs G_i^*, $i \in [\log \Delta]$: we define G_i^* to be the directed graph induced by $\mathcal{F}_i$, namely a directed graph on V, possibly with self-loops and multiple edges, which contains an edge uv, of length $d_{\text{M}}(u,v)$, whenever $v = X_{uj}^{(i)}$ for some $j \in [k]$.

For two nodes u, v and $i \in [\log \Delta]$, a (u,v)-path is *i-telescoping* if it consists of at most $\lceil \log \Delta \rceil$ edges such that (for every j) the j-th edge of this path is in graph G_{i+j}^* and takes us within distance $\frac{4}{3}\Delta/2^{i+j}$ from v. The reason we introduced (P4) is the following simple corollary:

Claim 4.1. *For any $r = \Delta/2^i$, $i \in [\log \Delta]$ and any two nodes u, v at distance at most $\frac{4}{3}\,r$ from each other, there exists an i-telescoping uv-path with high probability.*

The algorithm proceeds in stages $i = 0, 1, 2, \ldots, \lceil \log \Delta \rceil$. Informally, a given stage i handles distance scale $r = \Delta/2^i$. without loss of generality assume $\delta \leq \frac{2}{9}$, let δ be an integer power of two, and let $i_0 = i - \log \delta$. Each stage consists of three steps.

First step. In the first step, beacons are selected: each node u selects itself as an *level-i beacon* independently with probability close to $k_{\text{b}}/|B_u(\delta r)|$. Selection is implemented via random walks: we piggy-back on the construction of RRN. Specifically, we set aside k_{b} neighbors $X_{uj}^{(i_0)}$ in the i_0-th ring of u, and we 'select' if and only if one of these neighbor is u itself. The sole objective of beacon selection is to ensure that level-i beacons provide a good coverage on the scale of δr, and yet are relatively sparse on the scale of r; the former is used to prove accuracy, and the latter is used to bound load.

Claim 4.2. *For each node u, with high probability (a) there is at least one level-i beacon in $B_u(\delta r)$, and (b) there are at most $O(k_b)\,(8/\delta)^\alpha$ level-i beacons in $B_u(4r)$.*

Proof. Part (a) follows by (P5). Part (b) follows by Chernoff bounds since by Lemma 3.3 for any node $v \in B_u(2r)$ we have $|B_u(2r)| \leq (4/\delta)^\alpha\, |B_v(\delta r)|$. □

Second step. In the second step, level-i beacons declare themselves to other nodes via a special broadcast. This broadcast will involve at most $1 + \lceil \log \Delta \rceil$ types of packets, numbered from 0 to $\lceil \log \Delta \rceil$. Let $P_b(j)$ be a type-j broadcast packet from the special broadcast initiated by beacon b. Each beacon b initiates his broadcast by sending packet $P_b(i)$ to all its ring-$(i-1)$ neighbors. Each node stores a list of received broadcast packets (without duplicates). Suppose a given node receives a type-j packet $P_b(j)$. If $j < \lceil \log \Delta \rceil$ and this node has not seen such packet before, it sends packet $P_b(j+1)$ to all its ring-j neighbors; else it does nothing.

As a result of these broadcasts, each node u acquires the list S_{ui} of i-level beacons whose broadcasts it has received. For each beacon $b \in S_{ui}$ node u maintains upper and lower bounds on $d_{\text{M}}(u,b)$, denoted $D^+(u,b)$ and $D^-(u,b)$, and initialized to, respectively, ∞ and 0. They are updated in the third step using distances between beacons. We will show that eventually we construct good estimates on distances to all i-level beacons within distance $(1+\delta)\,r$ from u.

The special broadcast described above has the following properties:

Claim 4.3. *Consider the broadcast started by a level-i beacon b. This broadcast (a) reaches all nodes in $B_b(\frac{4}{3}\,r)$, (b) stays inside the ball $B_b(4r)$, (c) induces per-node load at most $O(k\,2^\alpha\,\log n)$.*

Proof. For part (a) note that for each node $u \in B_b(\frac{4}{3}\,r)$ the broadcast from beacon b follows each i-telescoping (b,u)-path; by Claim 4.1 at least one such path exists. For part (b), we prove by induction on j that packet $P_b(j)$ stays within distance $4r(1-2^{i-j-1})$ from beacon b. For part (c), recall that the in-degree of G^*_j is $O(k\,2^\alpha)$ (see the proof of Claim 3.4), and note that a given node can receive a given packet $P_b(j)$ only from its in-neighbors in G^*_{j-1}, and only once from each. □

Claim 4.4. *Consider the special broadcasts from all level-i beacons. The per-node load is at most $O(\log^4 n)(80/\delta)^\alpha$, and the total running time is at most $O(\log^5 n)(80/\delta)^\alpha$.*

Proof. By Claim 4.3(b) a given node u receives broadcasts only from beacons within distance $4r$ from u. By Claim 4.2 there are at most $O(k_{\mathrm{b}})\,(8/\delta)^\alpha$ such beacons, so by Claim 4.3(c) the load on node u is at most

$$L = O(k\,k_{\mathrm{b}}\,\log n)(16/\delta)^\alpha = O(\log^4 n)(80/\delta)^\alpha.$$

Let us bound the running time by $O(L\log n)$ via the following rather crude argument. Recall that a given node sends a given packet $P_b(j)$ at most once. Say the special broadcasts start at time 0. By induction on j, we claim that by time $t_j = O(L)(j-i+1)$ all packets $P_b(j)$ are sent and received. Indeed, if this is true for some j, then after time t_j a given node knows all packets $P_b(j+1)$ that it needs to send out, so it keeps sending them unless it needs to pause and receive some other packet. It will be done by time $t_j + O(L)$ since it can receive at most L packets. □

Remark. Assuming a minimal synchronization, namely that first all packets $P_b(i)$ are sent and received, then all packets $P_b(i+1)$, then all packets $P_b(i+2)$ and so on, a given node can aggregate all packets $P_b(j)$ that it sends (for a given j) into a very small number of packets. Then both the load and the running time become $O(k\,2^\alpha\,\log n) = O(10^\alpha\,\log^2 n)$.

Third step. In the third step (for $i > 0$), each level-i beacon b measures distances to all level-$(i-1)$ beacons that it knows, and for each $j \le i-2$ updates distance estimates to all level-j beacons that it knows. Specifically, each beacon $b' \in S_{(b,\,i-1)}$ beacon b measures distance to b', and receives from b' its distance estimates $D^\pm(b',b^*)$ for each level-j beacon $b^* \in S_{(b',\,j)}$. Then whenever this level-j beacon b^* also lies in $S_{(b,\,j)}$, beacon b updates its distance estimates $D^\pm(b,b^*)$:

$$\left\{\begin{array}{rcl} D^+(b,b^*) & \leftarrow & \min\left(D^+(b,b^*),\ D^+(b',b^*) + d_{\mathrm{M}}(b,b')\right) \\ D^-(b,b^*) & \leftarrow & \max\left(D^-(b,b^*),\ D^-(b',b^*) - d_{\mathrm{M}}(b,b')\right). \end{array}\right. \tag{6}$$

This completes the description of the algorithm.

A straightforward corollary of the update rule (6) is that $D^\pm$ are indeed upper/lower bounds:

Claim 4.5. *For any two nodes u,v we have $D^-(u,v) \le d_M(u,v) \le D^+(u,v)$ at any point in the execution of the algorithm.*

Note that in the last stage, every node is a level-$\lceil \log \Delta \rceil$ beacon with high probability. In particular, each node u forms bounds $D^\pm(u,b)$ for every beacon $b \in \cup_j S_{uj}$, which form the node label in triangulation. By Claim 4.3(b) and 4.2 the degree of this triangulation is

$$|\cup_j S_{uj}| \le O(k_{\mathrm{b}}\,\log n)(8/\delta)^\alpha = O(\log^2 n)\,(16/\delta)^\alpha.$$

It remains to show that our triangulation obtains the desired precision. To this end, let us first consider the distances to beacons:

Lemma 4.6. *Fix level $i \in [\log \Delta]$ and let $r = \Delta/2^i$. Then for each node u and each level-i beacon $b \in B_u((1+\delta)\, r)$ the bounds $D^{\pm}(u,b)$ are off from $d_M(u,b)$ by at most the additive factor of $2\delta r$.*

Proof. By Claim 4.2(a) for every level $j \in [\log \Delta]$ there exists a level-j beacon $b_j \in B_u(\delta r/2^{j-i})$.

First, recall that we assume $\delta \leq 2/9$, and so for each $j > i$ we have

$$d_{\mathrm{M}}(b, b_j) \leq d_{\mathrm{M}}(u,b) + d_{\mathrm{M}}(u, b_j) = r(1+\delta) + \delta r/2 \leq 4r/3.$$

By Claim 4.3(a) it follows that each b_j, $j > i$ receives the broadcast from beacon b. In particular, beacon b_{i+1} measures the distance to b.

Second, note that for each j

$$d_{\mathrm{M}}(b_j, b_{j+1}) \leq d_{\mathrm{M}}(u, b_j) + d_{\mathrm{M}}(u, b_{j+1}) \leq 3\delta r/2^{j+1-i} \leq r/2^{j-i},$$

so by Claim 4.3(a) beacon b_{j+1} receives the broadcast from b_j and, consequently, measures the distance to b_j. Now by induction on j we can show that each beacon b_j, $j \geq i+1$ forms bounds $D^{\pm}(b_j, b)$ that are at least as good as

$$D^{\pm}(b_j, b) = d_{\mathrm{M}}(b, b_{i+1}) \pm \sum_{l=i+1}^{j-1} d_{\mathrm{M}}(b_l, b_{l+1}).$$

Finally, recall that with high probability node u is a level-j beacon for $j = \lceil \log \Delta \rceil$. By the above equation,

$$D^{+}(u,b) \leq d_{\mathrm{M}}(u,b) + 2\sum_{l=i+1}^{j} d_{\mathrm{M}}(u, b_l) \leq d_{\mathrm{M}}(u,b) + 2\sum_{l=i+1}^{j} \delta\Delta/2^l \leq d_{\mathrm{M}}(u,b) + 2\delta r,$$

and similarly $D^{-}(u,b) \geq d_{\mathrm{M}}(u,b) - 2\delta r$ as required. □

Now we use Claim 4.3(a) and Lemma 4.6 to prove the desired accuracy.

Lemma 4.7. *For any two nodes u, v we have $D^{+}(u,v)/D^{-}(u,v) \leq 1 + O(\delta)$.*

Proof. Let us consider the distance scale i defined as the smallest i such that $r := \Delta/2^i \geq d_{\mathrm{M}}(u,v)$. By Claim 4.3(a) there exists an i-level beacon $b \in B_v(\delta r)$. Then $d_{\mathrm{M}}(u,b) \leq r(1+\delta)$, so by Lemma 4.6 both $D^{\pm}(u,b)$ and $D^{\pm}(v,b)$ are off from their respective true values by no more than the additive factor of $2\delta r$. It follows that

$$D^{+}(u,v) \leq D^{+}(u,b) + D^{+}(v,b) \leq d_{\mathrm{M}}(u,b) + d_{\mathrm{M}}(v,b) + 4\delta r \leq d_{\mathrm{M}}(u,v) + 6\delta r,$$

and similarly $D^{-}(u,v) \geq d_{\mathrm{M}}(u,v) - 6\delta r$. □

This completes the proof of Theorem 1.1.

References

[1] I. Abraham, Y. Bartal, H. T.-H. Chan, K. Dhamdhere, A. Gupta, J. Kleinberg, O. Neiman, and A. Slivkins. Metric Embeddings with Relaxed Guarantees. In *46th Symp. on Foundations of Computer Science (FOCS)*, pages 83–100, 2005.

[2] I. Abraham and D. Malkhi. Name independent routing for growth bounded networks. In *17th ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 49–55, 2005.

[3] N. Alon and J. Spencer. *The Probabilistic Method*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, New York, 2nd edition, 2000.

[4] M. Costa, M. Castro, A. Rowstron, and P. Key. PIC: Practical Internet coordinates for distance estimation. In *24th IEEE Intl. Conf. on Distributed Computing Systems (ICDCS)*, 2004.

[5] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A decentralized network coordinate system. In *ACM SIGCOMM*, 2004.

[6] M. Fomenkov, k. claffy, B. Huffaker, and D. Moore. Macroscopic Internet topology and performance measurements from the DNS root name servers. In *Usenix LISA*, 2001.

[7] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L. Zhang. IDMaps: A global Internet host distance estimation service. *IEEE/ACM Transactions on Networking*, 9:525–540, October 2001. Preliminary version in IEEE INFOCOM 1999.

[8] J. Friedman. A proof of Alon's second eigenvalue conjecture. In *35th ACM Symp. on Theory of Computing (STOC)*, pages 720–724, 2003. Full version available as *CoRR cs.DM/0405020*, 2004.

[9] C. Gavoille, D. Peleg, S. Perennes, and R. Raz. Distance labeling in graphs. *J. of Algorithms*, 53(1):85–112, 2004. (Preliminary version in *12th ACM-SIAM SODA*, 2001).

[10] C. Gkantsidis, M. Mihail, and A. Saberi. Random Walks in Peer-to-Peer Networks. In *IEEE INFOCOM*, 2004.

[11] J. Guyton and M. Schwartz. Locating nearby copies of replicated Internet servers. In *ACM SIGCOMM*, 1995.

[12] K. Hildrum, J. Kubiatowicz, S. Ma, and S. Rao. A note on the nearest neighbor in growth-restricted metrics. In *15th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 560–561, 2004.

[13] S. Hotz. *Routing information organization to support scalable interdomain routing with heterogeneous path requirements*. PhD thesis, University of Southern California, 1994.

[14] D. Karger and M. Ruhl. Finding Nearest Neighbors in Growth-restricted Metrics. In *34th ACM Symp. on Theory of Computing (STOC)*, pages 63–66, 2002.

[15] kc Claffy, T. Monk, and D. McRobb. Internet Tomography. *Nature*, Jan. 1999.

[16] D. Kempe, J. Kleinberg, and A. Demers. Spatial Gossip and Resource Location Protocols. *J. ACM*, 51(6):943–967, 2004. Preliminary version in *33rd ACM STOC*, 2001.

[17] J. Kleinberg, A. Slivkins, and T. Wexler. Triangulation and Embedding Using Small Sets of Beacons. In *45th Symp. on Foundations of Computer Science (FOCS)*, pages 444–453, 2004.

[18] C. Kommareddy, N. Shankar, and B. Bhattacharjee. Finding close friends on the Internet. In *12th IEEE Intl. Conf. on Network Protocols (ICNP)*, 2001.

[19] R. Krauthgamer and J. R. Lee. The Intrinsic Dimensionality of Graphs. In *35th ACM Symp. on Theory of Computing (STOC)*, pages 438–447, 2003.

[20] L. Lehman and S. Lerman. PALM: Predicting Internet network distances using peer-to-peer measurements. Technical report, MIT, January 2004.

[21] N. Linial and A. Wigderson. Course notes: Expander graphs and their applications. http://www.math.ias.edu/ boaz/ExpanderCourse/, 2002.

[22] M. Mendel and S. Har-Peled. Fast construction of nets in low dimensional metrics, and their applications. In *21st ACM Symp. on Computational Geometry (SoCG)*, pages 150–158, 2005.

[23] R. Motwani and P. Raghavan. *Randomized algorithms*. Cambridge University Press, Cambridge, 1995.

[24] T. E. Ng and H. Zhang. Predicting Internet network distance with coordinates-based approaches. In *IEEE INFOCOM*, 2002.

[25] M. Pias, J. Crowcroft, S. Wilbur, T. Harris, and S. Bhatti. Lighthouses for scalable distributed location. In *2nd Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.

[26] C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing nearby copies of replicated objects in a distributed environment. *Theory Comput. Syst.*, 32(3):241–280, 1999. Preliminary version in *9th ACM SPAA*, 1997.

[27] A. Sinclair and M. Jerrum. Approximate Counting, Uniform Generation, and Rapidly Mixing Markov Chains. *Information and Computation* , 82:93–133, 1989.

[28] A. Slivkins. Distance estimation and object location via rings of neighbors. In *24th Annual ACM SIGACT-SIGOPS Symp. on Principles Of Distributed Computing (PODC)* , pages 41–50, July 2005. Accepted to the special issue of *Distributed Computing* , currently under revision.

[29] A. Slivkins. Distributed Approaches to Triangulation and Embedding. In *16th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 640–649, Jan. 2005.

[30] K. Talwar. Bypassing the embedding: approximation schemes and compact representations for growth restricted metrics. In *36th ACM Symp. on Theory of Computing (STOC)* , pages 281–290, 2004.

[31] A. Vazquez, R. Pastor-Satorras, and A. Vespignani. Large-scale topological and dynamical properties of Internet. *Phys. Rev. E*, 65, 066130, 2002.

[32] B. Wong, A. Slivkins, and E. G. Sirer. Meridian: A Lightweight Network Location Service without Virtual Coordinates. In *ACM SIGCOMM*, 2005.

[33] D. Y. Xiao. The Evolution of Expander Graphs. BA Thesis, Harvard University, 2003. Available from *http://www.cs.princeton.edu/ dxiao*.

Appendix A: Expander graphs and Probability

Throughout this paper we use Chernoff Bounds, a standard result which says that the sum of bounded independent random variables is close to its expectation with high probability (e.g. see the textbook of Motwani and Raghavan [23] for the proof).

Theorem A.1 (Chernoff Bounds). *Consider the sum X of n independent random variables $X_i \in [0, y]$.*

(a) for any $\mu \leq E(X)$ and any $\epsilon \in (0, 1)$ we have $\Pr[X < (1-\epsilon)\mu] \leq exp(-\epsilon^2\mu/2y)$.

(b) for any $\mu \geq E(X)$ and any $\beta \geq 1$ we have $\Pr[X > \beta\mu] \leq \left[\frac{1}{e}(e/\beta)^{\beta}\right]^{\mu/y}$.

For an undirected graph, the *expansion* is defined as $\min \frac{|\partial(S)|}{|S|}$, where the minimum is over all nonempty sets S of at most $n/2$ vertices, and $\partial(S)$ stands for the set of edges with exactly one endpoint in S. We can generalize this definition to *weighted* undirected graphs, or, equivalently, to symmetric non-negative matrices: we just define $\partial(S)$ to be the total weight of all edges with exactly one endpoint in S. We can further extend this definition to directed graphs (non-symmetric matrices) by considering the weight of all edges leaving S.

For a pre-defined absolute constant, *expander* is an undirected graph whose expansion is at least this constant. Expanders are well-studied and have rich applications, see [21, 3, 23, 33] for more background. We will use the following standard result:

Theorem A.2 (Folklore). *Fix node set V. Suppose for each node u we choose three nodes independently and uniformly at random from V, and create undirected links between u and these three nodes. Then the resulting graph is an expander with high probability.*

See e.g. page 10 of [10] for the proof. We will actually need a stronger version where we select nodes from (and construct an expander on) any given subset Q of nodes, whereas we need the failure probability to be low in terms of n, not the size of Q. Hence we create $O(\log n)$ links per node instead of just three.

Theorem A.3. *Fix node set V of n nodes, and a subset $Q \subset V$. Suppose for each node $u \in Q$ we choose at least $3 \log n$ nodes independently from a near-uniform distribution on Q, and create undirected links between u and these nodes. Then the induced graph on Q is an expander with high probability.*

In Theorem A.2 and Theorem A.3 expanders have degree $O(\log n)$. We note in passing that for many applications it is useful to have constant-degree expanders. Indeed, such graphs exist; for instance, for large enough d a random d-regular graph is an expander with high probability [8].

A graph (V, E) induces a Markov chain on V as follows: for any edge $(u, v) \in E$, the transition probability $u \to v$ is set as $1/\deg(u)$. In particular, undirected graphs with low degree and high expansion gives rise to a Markov chains whose transition matrix has high expansion.

The following seminal result connects the mixing time of a Markov chain with the expansion of its transition matrix; we state it in a somewhat simplified form which is suitable for the purposes of this chapter.

Theorem A.4 (Rapid mixing, Sinclair and Jerrum [27]). *Consider an ergodic time-reversible n-state Markov chain with a uniform stationary distribution. Suppose that for every node the probability of stalling is at least $\frac{1}{2}$. Let γ be the expansion of the transition matrix. Then for any $k \geq O(\gamma^{-2})(\log n)$ and any initial distribution the k-step distribution of this Markov chain is near-uniform.*

The phenomenon when an n-state Markov chain achieves a near-stationary distribution in $O(\log n)$ steps is known as *rapid mixing*. The above theorem also extends to arbitrary stationary distributions.